1

查尔斯·西蒙尼

1948年9月10日，查尔斯·西蒙尼（Charles Simonyi）出生于匈牙利布达佩斯。上高中时，他开始接触计算机和编程，父亲安排他给一名从事计算机工作的工程师当助手，当时计算机在匈牙利屈指可数。

1966年，查尔斯高中毕业，同时也完成了他的第一个编译器。凭借开发编译器时积累的经验，他在丹麦哥本哈根的A/S Regnecentralen[1]公司谋得了一个职位。1968年，他离开丹麦进入美国加州大学伯克利分校学习，并于1972年获得理学学士学位，1977年获得斯坦福大学博士学位。

西蒙尼曾先后在加州大学伯克利分校计算机中心、伯克利计算机公司、ILLIAC 4项目和施乐PARC工作。自1981年以来，他一直供职于微软公司。在施乐公司，他开发了Alto个人电脑的Bravo和Bravo X程序。在微软，他组建了应用软件小组，并领导开发出Multiplan、Microsoft Word、Microsoft Excel等广受欢迎的应用软件。

在微型计算机世界的几乎各个领域，查尔斯·西蒙尼都打上了他的烙印，要么通过他自己的作品，要么通过影响和他共事的那些人。他谦逊而活泼，

① 丹麦第一家计算机公司，成立于1955年10月12日。（如无特殊说明，本书所有脚注均为译者注。）

脸上常挂着微笑，几乎能够就任何话题发表评论，不论是否与计算机相关。

我们跟查尔斯见过两次面，一次是在午餐时间，一次是在他的办公室，谈话内容无所不及，从Microsoft Excel的特性，到驾驶直升机，乃至现代诗歌的某些话题。他说话时带有很重的匈牙利口音，这已经成了查尔斯讲话和编程的独特标志。他每天几乎都穿同一身行头，褪色的牛仔夹克、衬衫和破旧的牛仔裤，看上去仍是一副20世纪60年代伯克利大学学生的模样，不过他的学识、举止和成就无不显示出他的过人智慧和丰富经验。

*　　*　　*

采访者：*你在匈牙利高中毕业之前就写了自己的第一个计算机程序，是吗？*

西蒙尼：是的。上高中时，我写了自己的第一个程序，还有第一个专业程序。我写的第一个程序是填充幻方，让每行、每列[①]的数之和均相等。我编程用的是一台古老的电子管计算机。一整个下午不停地推按钮才把程序输进那台机器。当天晚上，我头痛难耐，带着几大卷打印有80×80幻方的纸回到家里。那是1964年。

采访者：*说说你用过的第一台计算机？*

西蒙尼：那是一台俄制计算机，Ural II。它只有4K内存，支持40位浮点和20位操作指令。这台计算机只能用八进制机器码编程（没有汇编器）。我写了几千行八进制机器码。

这台计算机的操作全部通过控制台完成，你需要自己动手，跟它进行一对一的交互。程序员不必站在一旁等待另一位计算机操作员执行一批卡片。从这个角度看，Ural II酷似个人计算机，因为除了机器和你，不用其他人介入。就4K的内存和缓慢的速度而言，它跟1974年推出的Altair非常相似。1964年Ural II带给我的兴奋就和1974年Altair带给比尔·盖茨的兴奋一样。

显然，Ural II在某些方面有别于个人计算机。Ural II体积庞大，要占用一间很大的房间，输入和输出的方法极为原始——主要是通过控制台开关。控制台看起来像一台老式收银机，上面有整整六列开关，右侧有一个输入键。每一列有8个键，编号从0到7。输入数字的方式同操作收银机差不多。因此，

① 严格来说还包括两条对角线。

要输入2275，你需要依次拨动2、2、7、5这几个键。不小心按错的话，只要还没有按下右侧的输入键，都还可以修正。这种操作非常提神，因为它会伴有大量噪音。每次拨动开关都会发出响亮的喀哒声，每当清掉按键时——这全靠机械完成——所有按键一下子同时释放，伴着巨大的啷啷声。

采访者：你的第一个专业程序是什么样的？

西蒙尼：我的第一个专业程序是为一种非常简单、类似FORTRAN的高级语言写的编译器。我把它作为一项创新成果卖给了政府部门，并得到一大笔钱，不过我一分也没花，因为不久之后我就离开了匈牙利。

机遇出现在布达佩斯的一次交易会上，我见到几位从事计算机工作的丹麦人。我跟他们接洽，了解到他们新机器相关的大量信息。在随后一次交易会上，我带上自己事先准备好的一个小演示程序，它能准确反馈任意时刻机器正在分析长表达式的哪一部分。我拜托其中一人把这个程序带回丹麦，拿给他们的主管看。他们肯定很喜欢这个程序，因为他们给了我一份工作。我就这样离开了匈牙利。

我在丹麦干了一年半的编程，攒够了钱去加州大学伯克利分校求学。在校期间，我进入伯克利计算机中心当程序员，挣的钱刚好也够付学费的了。

在伯克利上学时，我写了一个很不错的SNOBOL编译器。有个计算机科学教授，叫巴特勒·兰普森，非常喜欢这个编译器，他还让计算机科学专业的学生在课堂上使用它。后来，他跟另外几个教授一起创办了伯克利计算机公司，我便在那家公司谋得一份工作。伯克利计算机公司倒闭后，核心成员都去了施乐PARC。

采访者：你的编程风格主要受谁的影响？

西蒙尼：影响主要来自两方面——一位匈牙利工程师，一台我在丹麦工作时用的计算机。我在匈牙利的导师是一位使用Ural II计算机工作的工程师。我像个狂热的追星族，卑躬屈膝外加免费跑腿，以换取别人容许我待在一个我本不该待的地方。这不是孩子待的地方。它是全匈牙利（也许）仅有的五台计算机中的一台，被看作重要资产。

采访者：你是怎么个卑躬屈膝法？

西蒙尼：我父亲是电子工程学教授，这个工程师是他的学生。我猜是我父亲

托他帮忙让我进去的。我也尽量让自己能派上用场。我先是给他带午饭，后来帮他拿东西递家伙，最后我主动提出帮他们守夜，看管机器。

他们一到晚上就把计算机关掉，到第二天早上再打开。开关真空管时，电热丝加热或冷却很容易损坏。这台机器有2000个真空管，每次打开时都会坏掉一个。因此，他们上班后的第一件事就是先花一个小时找出那个坏掉的真空管。我在那里守夜的话，计算机就可以一直开着，他们也不用浪费那一个小时。于是，在晚上看管机器的时候，我也就可以用这台计算机了。

总之，我和这位工程师成了好朋友。他是个数学天才。我早年学到的许多技巧都是他教的，有的是关于算术思考，有的是关于符号问题。

另外，那台丹麦计算机对我影响也很大。当时，它拥有的也许是世界上最好的Algol编译器，Gier Algol。去丹麦之前，我已经把这个编译器的全部代码清单研究了个遍。它全都是用机器语言写成的，因此我既学了机器语言编程，又学会了从美学层面上思考编译过程。这个编译器的设计者是彼得·诺尔（Peter Naur）[①]。语法等式巴科斯–诺尔范式（BNF）中的字母N就取自他的名字。我对这个程序知根知底，至今仍记忆犹新。

举个例子，我在伯克利上学时写的SNOBOL编译器只是这个程序的变体。我觉得Gier Algol程序现在仍在我脑海中，也影响着我的编程风格。我总是问自己："如果这是Algol编译器的一部分，他们会怎么做呢？"这个程序真是精妙无比。

有一点我印象很深，就是他们倒着扫描源代码文本的做法。在某些情况下，如果你倒着做事情，之前显得很复杂的问题突然之间会变得非常简单。例如，解析前向引用（forward reference）可能很难。要是倒着扫描，它们就变成了后向引用（backward reference），很容易解析。只要从新的角度看待程序，原本可能很难解决的问题也会变得容易解决。这个Algol编译器处处是玄机。

采访者：*你是怎么进入微软的？*

西蒙尼：决定离开施乐之后，我开始四处打探。我请鲍勃·麦特卡尔夫（Bob Metcalfe）共进午餐。鲍勃是以太网发明人，3Com公司的董事长和创始人，

---

① 出生于1928年10月25日，是丹麦计算机科学先驱，为ALGOL 60编程语言的创建和定义做出了很大贡献，并因此获得2005年图灵奖。他是目前唯一一个获得图灵奖的丹麦人。

早我两年离开施乐。他给了我一张名单，上面列有我应该去找的人。名单上比尔·盖茨排在第一位。谁排在第二我记不清了，因为除了比尔·盖茨我没再找过其他人。

采访者：*编程是一种技巧或技能吗？*

西蒙尼：什么是编程？人们对此一直各持己见。有人说它是科学，有人说它是艺术，还有人称之为技能或手艺。我认为这三方面兼而有之。我们喜欢说它蕴含大量艺术成分，但是我们都知道它里面更多的是科学。

孩子们在学校里学习数学，高中毕业时，他们会以为数学就是加法和乘法，甚或代数和微积分。其实，算术，即使简单如加法的运算，背后也有令人难以置信的科学理论作支持。

计算机编程背后也有大量科学理论作支持。例如，哥德尔定理的数学证明冗长而复杂，但是如果借用计算机科学的图灵定理，证明起来不费吹灰之力。信息理论和计算机科学其他领域对数学影响巨大，反之亦然。

编程包含有大量科学，同时，它也有点像手艺。实际上，在许多人看来，编程是一项复杂的技能，这跟工具制造很像，需要精雕细琢。我认为，只要将科学、艺术和技能这三者拿捏得恰到好处，你就能取得一些引人瞩目的成绩。

采访者：*你觉得编程的哪部分可以视作艺术？是用户界面设计吗？*

西蒙尼：在我看来，编程显然有审美的一面，对用户界面而言，不仅设计中存在，甚至连外观也不例外。当你看到那些丑陋的屏幕时，程序员在艺术上的不足便一览无遗。在其他方面，计算机编程也堪称艺术，正如高能物理也可视作艺术一样。

采访者：*审美是只关乎用户对程序的感觉，还是它也直接影响到其他程序员分析该程序并探究其编写方式？*

西蒙尼：绝对也会影响的，毋庸置疑。我觉得代码清单和计算机自身的美感一直让我陶醉其中。

例如，那台俄制机器看起来像是科幻小说里的计算机，因为机器里的每个触发器（存储1比特信息的开关装置）都有一个小小的、橙色的老式气体放电灯。数以百计的橙色小灯在玻璃门和柜子后面不停闪烁。机器整个生命

的脉动仿佛就在眼前。

那台丹麦计算机是件精美的家具。它的大小与旧式的衣橱相当。计算机正面有3扇柚木门。有一次，我看到有个美国来的主管半信半疑地盯着机器，就因为它是用柚木嵌板的。它甚至还有一个丹麦现代风格的桌台。整台机器散发着迷人的柚木味道。

伯克利计算机个头非常大，大约有6米长，1.8米高，0.6米深。它隐藏在完全漆成黑色的混凝土穹顶里。它放在穹顶里打着聚光灯的样子看上去有点像电影《2001太空漫游》[①]里的黑色独石。

采访者：当你分析某个程序时，你认为什么样的代码清单或算法结构在审美上是优美或悦人的？

西蒙尼：我觉得代码清单带给人的愉快同整洁的家差不多。你一眼就能分辨出家里是杂乱无章（比如垃圾和没洗的碟子到处乱扔）还是整洁如新。这也许意义不大。因为光是房子整洁说明不了什么，它仍可能藏污纳垢！但是第一印象很重要，它至少反映了程序的某些方面。我敢打赌，我在3米开外就能看出程序拙劣与否。我也许没法保证它很不错，但如果从3米外看起来就很糟，我敢保证这程序写得不用心。如果写得不用心，那它在逻辑上也许就不会优美。

不过假定它看上去不错，然后你打算继续深入。理解程序的结构要困难得多。在结构因何优美的问题上也是见仁见智。纯粹主义者认为，只有那些按照极其严格的数学方式来使用某些很简单的构造的结构化编程，才是优美的。就20世纪60年代之前的情况而言，这种反应非常合乎情理，因为当时程序员并不知道结构化的概念。

不过在我看来，即使程序不遵循这些概念，只要它们有其他可取之处，也可以算是优美的。这就像拿现代诗歌和古典诗歌比较。我觉得古典诗歌很棒，你可以欣赏它。但是你不能只欣赏古典诗歌而无视其他。另外，这也并不意味着，只要在纸上胡乱写上一些字，称之为诗歌，就有了美。但是，如

① 2001: A Space Odyssey，一部颇具影响力的美国科幻电影，1968年上演，由斯坦利·库布里克导演。故事根据科幻小说家亚瑟·克拉克撰写的多篇短篇小说的部分内容改编而成，包括1950年的短篇《前哨》，并间接引用《童年末日》中克拉克大力提倡的人类优越论为其题材。

果代码有一些可取之处，我不认为非得是数学意义上的结构化才称得上优美。

采访者：别人读几段你的源代码，有没有可能断定“这代码是查尔斯·西蒙尼写的”？

西蒙尼：噢，是的，毫无疑问。是不是我本人写的可能很难分辨，但有一点是确定无疑的：只要看了代码，你就能知道它是不是我的团队写的，或者是不是受我的影响写的。这是因为我从1972年起写的代码都遵循特定的命名规范，许多人称之为“匈牙利命名法”。你一眼就能分辨出哪些代码是受我的影响写出来的，包括Microsoft Word、Multiplan和Bravo，以及其他许多遵循这些规范写成的程序。

采访者：你提到的“匈牙利命名法”是指什么？

西蒙尼：称它为“匈牙利命名法”是个玩笑。你知道，如果有人说“这对我来说就是希腊文”，这表示他们看不懂，因此也可能它就真是用希腊语写的。这里的“匈牙利”是句反话，因为这些命名规范其实是要让代码更易读。这个玩笑说的是程序看起来这么难读，说不定真是用匈牙利语写的。其实这套规范能够很好地控制程序中所有变量的命名。

要是分解一个程序，把它放进磨床，然后对碎片进行分类，你就会发现程序的大部分都是名字。写下“apples + oranges”，你会发现名字“apples”有6个字符，运算符“+”只有1个字符，名字“oranges”有7个字符，一共14个字符，只有一个字符即加号与运算有关。因此，对我来说，要起到作用或有所改进，合乎逻辑的做法就是尽力完善程序的主要部分，也就是名字。“匈牙利命名法”是一种根据变量的属性自动为其创建名字的命名方法。这跟人们把当裁缝（tailor）的叫做泰勒（Taylor）以及把当铁匠（blacksmith）的叫做史密斯（Smith）非常相似。

因此，面对一个具备某些属性的结构，不要随随便便地取个名字，然后让所有人去琢磨名字和属性之间有什么关联，你应该把属性本身用作结构的名字。这种方法有很多优点。首先，造个名字很容易，想到那些属性时，把它们写下来，名字自然就有了。第二，它很容易理解，因为当你读到某个变量时，从名字本身就能了解到与属性有关的大量信息。这些属性会越来越多，因此很难简明地描述它们。为此“匈牙利命名法”引入了一种缩写符号，以

很小的空间就能展现具体属性。当然，这在不知情的人看来完全是一团乱麻，那个玩笑就是这么来的。

有些人认为，如果他们可以读出代码里的每个字，那么程序就是可读的。实际上，这种意义上的可读性并不可取。没有人会拿着代码清单，站到演讲台上大声朗读程序。关键在于理解。只是能阅读单词并发出音来，这毫无用处。当人们看到采用“匈牙利命名法”的代码清单时，他们发现这些词很难念，可能就会认为代码不是可读的。但实际上，由于名字和属性之间存在关联，它更容易理解，也更便于沟通。那些使用匈牙利命名法编程的人，即使在离开我的部门之后，仍会继续使用它。这种命名法已经打入苹果电脑、3Com及其他许多公司。

采访者：*下面说说你创建程序的整个过程。是否存在适用于所有程序的过程？*

西蒙尼：当然。严格来说，对编程而言，我认为我们应该知道自己想要做什么。如果不知道，那么有一个过程确实是解决各种问题的必经之路，那就是要弄清楚：我试图做什么？目标是什么？

打个比方，我想开发一个菜单驱动的文本编辑器，要求响应速度快，并且提供拼写检查器等。在开始真正编程之前，我需要先弄清楚最终产品。有时候，目标的选择取决于我都掌握了哪些技巧。以Bravo为例，这个程序是以算法为导引的。巴特勒·兰普森描述了两个很有意思的算法，于是我们试图围绕这些算法来编写这个编辑器，以充分利用这些算法。此外，J. 斯特罗彻·摩尔（J. Strother Moore），就是Boyer-Moore字符串查找算法的Moore，在文档编辑方面有几个很有意思的算法。于是我们决定：“嘿，这个编辑器要包含摩尔编辑算法、兰普森的屏幕更新算法还有两个缓存。”等到对目标有充分的把握之后，我才会开始真正的编程。我调整姿态，关上房门，并且大声宣布：“现在我要开始编程了。”

采访者：*当你调整好状态真正开始编程时，第一步会做什么？*

西蒙尼：编程的第一步是想象。就是要在脑海中对来龙去脉有极为清晰的把握。在这个初始阶段，我会使用纸和铅笔。我只是信手涂鸦，并不写代码。我也许会画些方框或箭头，但基本上只是涂鸦，因为真正的想法在我脑海里。我喜欢想象那些有待维护的结构，那些结构代表着我想编码的真实世界。

一旦这个结构考虑得相当严谨和明确，我便开始写代码。我会坐到终端前，或者换在以前的话，就会拿张白纸，开始写代码。这相当容易。我只要把头脑中的想法变换成代码写下来，我知道结果应该是什么样的。大部分代码会水到渠成，不过我维护的那些数据结构才是关键。我会先想好数据结构，并在整个编码过程中将它们牢记于心。

采访者：这是最重要的一步吗？

西蒙尼：当然，这是最重要的一步：最优算法的知识当属科学，结构的想象则是艺术。这些算法的细节，以及编写高效代码实现这些结构的转换，是编程像手艺活的一面。从技术上讲，这就是所谓维护结构的不变性。编写代码以维护不变性是相对简单的技艺，不过这需要非常用心并辅之以大量训练才能练就。

采访者：你对编程感到过厌倦吗？

西蒙尼：是的。

采访者：编写程序的过程是痛苦的还是快乐的？

西蒙尼：两者兼而有之。假装每时每刻都很快乐是做作。就像运动员所说的："要是没受伤的话，肯定是你还不够努力。"二十年后，我已经体会不到刚开始编程一两年时的那种新鲜感。当然，有时我仍会有这种感觉，只不过不像以往那样常有，这是没办法的事。

采访者：你每天都有固定安排吗？你每天都编程吗，或者你会先把问题放一放，然后集中一周时间搞定它？

西蒙尼：我不是每天都有机会编程。我不用特意把问题放一放，因为总会有人打断我。我一般晚上编程，白天总是被打断。

采访者：晚上你会到办公室还是在家工作？

西蒙尼：我就在办公室工作。我住得很近，非常方便。来办公室就像进自己家另一个房间。我不会窝在家里编程的，来办公室也就是两分钟的事儿。

采访者：你如何管理手下的程序员？你觉得现在自己做管理多过编程吗？

西蒙尼：我两样都做，目前还是编程多一些。开发Bravo时，对程序员的管理非常非常直接。有一次，我其实写了一份极为详尽的工作指令，也就是所谓的元程序。这差不多就是个程序，只不过是用非常非常高级的语言写的。我们从斯坦福大学找了两个机灵鬼作为“试验对象”。他们写的程序完全符合我的要求，这样我们实现了双赢：首先，对我来说，用这种非常高级的语言工作更容易，本质上是在对这些人进行编程；其次，他们真正弄清楚了这个程序，效果远远好过我直接交给他们写好的代码清单，并叮嘱他们仔细研读这个程序。他们掌握了这个程序，因为这是他们写的。瞧，每个人都可以宣称自己写了这个程序。这个程序是我写的，也是他们写的。真是太棒了！我认为管理的最佳方法是言传身教，经常复审代码。我们一直坚持开展代码复审。

采访者：让多名程序员开发一个程序，开发速度会更快吗？

西蒙尼：不一定。编写同一个程序的人员越多，人均产出的实际代码量越少。结果，总的代码产出一开始会更多，之后实际上可能会减少。以两个人为例，也许单位时间只能多写百分之五十的代码。

顺便提一下，代码的效率还会随着开发同一个程序的人员数量的增加而有所降低。最高效的程序往往是一个人写的。唯一的问题是，它可能需要写上一辈子，而这显然是无法接受的。因此你需要找上三五十个，甚或好几百个人开发一个项目。

采访者：你能预估编写一个程序要用多长时间吗？

西蒙尼：预估编写程序要花的时间难度很大。之所以难度很大，原因多种多样。这并不意味着我们就不用尽全力预估，因为预估时间可能用处很大，就像天气预报不仅有经济效益还有其他好处一样。

真正的好程序会永远存在，写起来永无止境，至少只要硬件存在，程序就会存在，甚至更长久。当然，只要Alto计算机存在一天，Bravo就不会消失。编写Bravo的是两个暑期实习生。夏天结束时，其中一人走了，另一个人留了下来。第一个发布版本大概用了3个月。在约5年时间里，一共发布了14个版本。

Multiplan也是大致如此。当你想到Multiplan存在于Microsoft Excel中，

就明白Multiplan将不断延续下去。而Macintosh上的Microsoft Excel也不会是这一链条中的最后一个程序。它会在Windows上延续下去。

采访者：开发Bravo时，你是否认为施乐Alto计算机会成为所有人的选择？

西蒙尼：是的，我当时太天真了。不过，Alto后续机器正逐渐成为每个人都能用的机器，由此可见我的看法并没有错。从某种意义上说，Macintosh计算机和Windows程序都是后继者……（说到这里，西蒙尼停下来接电话，草草地说了几句便接着回来和我聊）。来电话的是汤姆·马洛伊（Tom Malloy），我刚才提到的其中一个暑期学生，留下来的就是他。我有一年没跟他聊过了。他后来去了苹果公司，为Lisa电脑开发编辑器程序。

采访者：你为什么会写程序？你把它看作是工作、职业还是挣钱的手段？这是你与生俱来的能力吗？

西蒙尼：应该是兼而有之吧。我年轻时还算有点天分。即使是在我不懂编程的时候，我就知道与编程有很大关系的一些东西。记住那些复杂的事情对我来说轻而易举。随着年龄的增长，这变得越来越困难。想象场景也不再那么清晰。

采访者：为什么想象场景变得不再那么清晰？

西蒙尼：可能只是年纪大了，思维方式也跟着变了。现在，如果试图清晰明确地想象出包含二三十个组件的东西，我就必须全神贯注，甚至还可能会头疼。换作年轻的时候，我可以想象一座有20个房间的城堡，每个房间里有10个不同的物体，这都不在话下。不过现在我做不到了。现在我更多的是依赖早年的经验进行思考。我看到的是一团团未成形的云，不是像明信片画面那样清晰的图景。不过我的程序确实写得更好了。

采访者：成为优秀程序员有什么套路可循吗？

西蒙尼：恐怕没有。

采访者：这种才能是天生的还是来自后天教育？

西蒙尼：挑选合适的人并培养成优秀程序员，有不少套路可循。我们招揽有天分的员工。我不清楚他们何以有如此才华，我也不用去弄清楚。但是他们

确实很有才。在此基础之上，环境可以起到很大作用。

进入公司第一天，程序员就会拿到几本书。其中一本是数学家乔治·波利亚写的《怎样解题》。（西蒙尼边说边从他办公桌旁的书柜里取出那本书，翻到某一页。）这两页很重要。这本书的其余内容就是基于这两页展开的。这就像一张问题求解的检查单。这是起飞前、起飞和着陆检查单。它不会教你如何飞行，但是如果不照做，即使你已经懂得怎么飞行，也有可能会坠机身亡。

求解问题时，我们遵循以下四个步骤：首先理解问题，然后拟定计划，接着执行计划，最后回顾整个过程。这样的书我们大概有四本，我觉得我们能使程序员比刚加入公司时变得更加优秀。

采访者：你怎么看待将来程序员的作用？

西蒙尼：如果你是在问我们会不会像以前的物理学家那样自高自大，天知道！任何一门学科只要硕果连连，从事那门学科的人似乎总会飘飘然：“我们早就知道自己真的很聪明。”然后，他们摩拳擦掌，想要解决其他领域的问题。

这让我想到1945年之后的物理学家。他们说：“我们全都搞定了！现在让我们四处看看。”看到生物学与控制论，他们认为：“这些研究大脑的家伙什么都不懂。他们甚至不知道记忆是怎么储存的。这也不足为奇，谁叫他们都是呆瓜呢。现在，让**我们**来研究研究。**我们**会搞定它的。我们将运用海森堡方程、量子力学，或者以前派过用场的随便什么玩意，我们会把它应用到大脑研究中，接下来就等着奇迹出现吧。”

这有时行得通，有时行不通。天知道。也许计算机科学将会帮助破译DNA，而不是只提供工具。破译DNA可能会成为黑客的终极梦想。

采访者：你觉得将来程序的编写方式会有重大改变吗？

西蒙尼：我认为未来的计算机会比今天的更高效，但我不觉得会有什么大的差别。我不知道第6代或32代计算机会不会做到一些完全不同的或是很了不起的事情。我对鼓吹多妙多好的新方法格外警惕。光从我们现有的方法来看，我就能看出不少改进余地。我更信赖现有的方法，不是因为我保守，而是因为我知道自己至少不会失去既有好处。

我总是担心，当这些所谓难以置信的新好处到来时，以往所有的好处也

将不复存在。然后，它会让你陷入两难境地，你必须自行判断境况是否更好。我喜欢有把握的胜利。我敢断定，改善我们现有的东西，保留既有好处并消除弊端，胜过引入新好处新弊端兼而有之的新玩意。不过也许我是错的，到时我会第一个加入到新事物的行列中。我坚信事情会发生剧变，变得更好，不过这需要时间。

采访者：为什么还要等这么长时间？

西蒙尼：因为必须先等大量愚蠢的想法消逝。这就是为什么进步需要时间。首先，新想法必须不断演变；其次，阻碍进步的坏想法必须消亡。历来就是如此。即使是相对论和量子力学，好的想法也必须经历很长时间才能成形。然后，旧物理学的既得利益者会慢慢消失。

采访者：可以举个例子吗？

西蒙尼：要是提到人们普遍厌恶的打孔卡之类的东西，我估计不会有多大效果。因此，我还是得挑些大多数人相信的东西。我认为“简单崇拜”，即以简单本身作为追求目标的观念，值得高度怀疑。多年来，这个观念一直诱使我们关注那些回报最快的问题。但它只是一种手段。我认为，计算机科学，连同其他所有数理科学（数学、物理和现代分子生物学等），都将通过理解非常复杂的现象而不断改进。数学已经发现了一些复杂的基本对象，在这方面也处于领先地位。有一类这样的对象其传统名称是“简单群”，这颇具讽刺意味地反映出“基本”等于“简单”的旧观念。不过，这两个含义也许并不相同。在计算机中，只是喋喋不休地大谈简单性，我们在实际的人工智能、用户界面和语言等领域也许就会毫无进展。

采访者：不编程的时候，你都做些什么？还有其他兴趣爱好吗？

西蒙尼：还有其他不少有意思的事情，我也乐此不疲。我对埃及象形文字略知一二。学习其他语言、旅行和观察世界都是很不错的活动，我不介意做这些事情。我还持有私人旋翼飞机（直升飞机）飞行员执照。

我不觉得编程要比其他事情重要得多。但是如果你从做事的角度来看，它就变成另一回事了。实际上，比起单纯的编程，正是这份业务工作让我忙得不可开交。我非常愿意在工作过程中做编程。

采访者：你是否认为自己更愿意把时间用来完成编程这项业务工作？

西蒙尼：不。我只是说我编程是因为它是工作。不只是因为我喜欢编程，而且因为我喜欢这项工作。我不会一写代码就雀跃不已，说："嘿，我又写了一行代码，真是太开心了；写得越多，快乐越多。"绝非如此。这行代码我也许已经写了10遍。只是不断重复键入，以致弄伤手指，可能会非常倒胃口。因此当我编程时，那么做的原因在于它是工作的一部分，这份工作才是我想做的。

抽象的编程和业务的编程之间，主要区别在于后者目的非常明确。否则，它不过是像下棋一样的抽象活动，游戏终了，棋子乱成一堆，游戏就此结束。程序完成时，有人会使用它，我看到他们喜欢它，自己也会从中获得满足。他们中有些人甚至为它付了钱，我能分到其中一部分，可以用这些钱到埃及旅游，或者飞半小时直升飞机。顺便提一下，驾驶飞机跟编程项目非常相似：起飞和降落很刺激，驾驶过程可能会非常累人，而飞机可能会随时解体。

采访者：*你怎么看待自己同时代的其他程序员？*

西蒙尼：我非常敬重那些影响过我的人，我非常敬重他们。不过我也非常尊重现在跟自己共事的人。

采访者：*你跟其他开发过大型程序的程序员有无来往？你会和他们交流想法吗？*

西蒙尼：我非常看重竞争。我有幸在两次展会上遇见鲍勃·弗兰克斯顿和丹·布兰克林（这两位是Software Arts公司共同创始人，世界首套电子表格软件VisiCalc的创作者）。我还遇到过乔纳森·萨奇（莲花公司[①]创始人之一）。但遗憾的是我们来往并不多，这些程序员来西雅图的机会也不多。布鲁斯·阿特维克（Bruce Artwick，微软飞行模拟软件Flight Simulator 的编写者，被称为模拟飞行之父）有时会来造访，还有苹果公司的人，比如比尔·阿特金森（Bill Atkinson，Lisa电脑程序员之一，后来为苹果Macintosh电脑开发了MacPaint程序）——在我眼里他是最棒的——和比尔·巴奇（Bill Budge），为美国艺电公司开发了《弹珠台制造机》[②]游戏）。这些家伙都很出色。

---

① Lotus Development Corporation，1995年被IBM收购后更名为Lotus Software。
② Pinball Construction Set'PCS，一种新的游戏类型，用户可以用它制造虚拟的弹珠台街机。

我们没有太多东西可以高谈阔论。我们都很有默契，说上三两句话就能心领神会。我知道，这些家伙只要开口说话，就会知道自己在说什么。因此当他真正开口说话、也确实知道自己在说什么时，就没什么让人意外的了。而且既然我也知道自己在说什么，说的又是同一件事，那又何必说呢，是吧？这就像笑话专场，听众围坐一圈，说笑话的人甚至用不着讲笑话。他们只要报个笑话的编号，大家就笑翻了。

要是能和这些人一起共事，那就太棒了，只可惜我们却是商业上的竞争对手。我觉得我们聚到一起可以成就一番伟业。也许有一天火星人入侵地球，逼不得已，我们必须实施计算机曼哈顿计划[①]。我们全都会被送到新墨西哥州并肩作战。天知道。

续写传奇人生

西蒙尼从1981年进入微软，直到2002年离开，当时他在微软公司的头衔是应用开发总监、首席架构师。在微软期间，西蒙尼招聘和管理的开发团队创造了很多最畅销的软件，包括Microsoft Word、Microsoft Excel等。西蒙尼于2002年创办了Intentional Software，目前担任该公司主席和CTO。这家公司的宗旨是创造能加速软件设计的技术，让商务人士即使不熟悉电脑术语，也能清楚地描述需求。

在工作以外，西蒙尼表现出对航天旅行的极大兴趣，并于2006年9月在俄罗斯星城接受训练。（星城是俄罗斯加加林宇航员培训中心的别称。）2007年4月7日，他与两位俄罗斯宇航员一起搭载联盟TMA-10飞船前往国际空间站，并于21日返回地球。4月9日到达国际空间站的时候，西蒙尼说："黑暗天空中的一切都令人惊叹，非常非常激动人心。就像一个巨大的舞台布景，有许多不可思议的歌剧或现代剧的奇妙演出。当我说我彻底折服的时候，就是现在这个样子。"

两年后，即2009年3月，西蒙尼再次进行了太空旅行，重游国际空间站。

① Manhattan Project，美国陆军于1942年6月开始实施的利用核裂变反应来研制原子弹的计划，主要在新墨西哥州沙漠地区的一处绝密研究中心进行。